

A METHOD AND SYSTEM FOR AUTOMATING THE CONFIGURATION OF A STORAGE AREA NETWORK

BACKGROUND OF THE INVENTION

1. Technical Field

The present invention relates generally to an improved distributed storage system and in particular to a method and a system for configuring a storage area network. Still more particularly, the present invention provides a method and a system for automating the configuration of a storage area network.

2. Description of the Related Art

The primary model of computing has been the mainframe computer with satellite terminals, but this model is being replaced with a client-server model of computing. The client-server model is distributed and may involve a plurality of servers and a plurality of clients, each a powerful workstation in its own right.

Early client-server models inherited from the mainframe model the concept of dedicated storage for each server. There are several major problems associated with using dedicated storage in this environment. Every time a client or a server needs to access information that is not available locally, the computer must issue a request for data to a peer machine. It takes time to process these requests and, if there is a heavy volume of non-local access, a bottleneck may result. In addition, one or more servers may have requirements for data storage that exceed their capacities while other servers may only be using a fraction of their available data storage.

A Storage Area Network (SAN) is a relatively new innovation for the client-server model based on a data-centric infrastructure with a common storage pool. A SAN works well provided there are persistent communications links. Distributed networks, such as the Internet, are constructed with persistence in mind. In other words, the loss of one or several communications links will not shut down the network and an alternative path through the network can almost always be found.

One or more configuration files describing the details of the SAN need to be maintained at each server in the network. A network may contain tens or hundreds of servers and servers may be added or removed on a continual basis. Each configuration file on every server connected to the network must be changed as the machines on the SAN change. It is far too tedious and error prone to manually perform reconfiguration on a server-by-server basis. A SAN may include a variety of operating systems, such as, for example, Solaris™, a version of UNIX used by computers from Sun Microsystems, AIX™, a version of UNIX used by computers from International Business Machines, and Windows NT™, an operating system from Microsoft that is used by a variety of workstations. Transferring data between different operating systems must be done very carefully since the file system formats are not compatible. Transferring data in the wrong format could result in loss of the data and damage to the file system. It only takes a single incorrect entry of the operating system type to cause this problem to occur. If data is being entered and maintained manually on a server that is part of the SAN, this type of error is very probable. Therefore, it would be advantageous to have a master

SUMMARY OF THE INVENTION

The present invention provides a method and system for the development and maintenance of a single configuration file for a storage area network combined with an "intelligent" script which places server configuration files into the appropriate directories for servers that are part of a storage area network. On a server by server basis, each server is configured in accordance with the particular operating system the server itself is using. Once this is accomplished, each server added or removed from the storage area network will be processed in a similar way without the need for reconfiguration of the storage area network as a whole.

BRIEF DESCRIPTION OF THE DRAWINGS

The novel features believed characteristic of the invention are set forth in the appended claims. The invention itself, however, as well as a preferred mode of use, further objectives and advantages thereof, will best be understood by reference to the following detailed description of an illustrative embodiment when read in conjunction with the accompanying drawings, wherein:

Figure 1 is a pictorial representation of a distributed data processing system in which the present invention may be implemented;

Figure 2 is a block diagram illustrating a data processing system that may be implemented as a server in which the present invention may be implemented;

Figure 3 is a block diagram illustrating a data processing system that may be implemented as a client in which the present invention may be implemented;

Figure 4 is an exemplary illustration of a network containing a storage area network (SAN) in accordance with a preferred embodiment of the present invention;

Figure 5 is a flowchart illustrating the automatic configuration of a server within each SAN in accordance with a preferred embodiment of the invention;

Figure 6 is a flowchart illustrating the transfer of configuration files for an individual server within a SAN in accordance with a preferred embodiment of the invention; and

Figures 7A-7D is an exemplary shell script for updating configuration files in accordance with a preferred embodiment of the invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

With reference now to the figures, and in particular with reference to Figure 1, a pictorial representation of a distributed data processing system is depicted in which the present invention may be implemented.

Distributed data processing system 100 is a network of computers. Distributed data processing system 100 contains network 102, which is the medium used to provide communications links between various devices and computers connected within distributed data processing system 100. Network 102 may include permanent connections, such as wire or fiber optic cables, or temporary connections made through telephone connections.

In the depicted example, servers 104, 114, 116 and 118 are connected to network 102. Storage units 106 and 122 are also connected to network 102, providing backup support for any or all of servers 104, 114, 116 and 118. Storage unit 120 provides dedicated backup support for server 104. In addition, clients 108, 110 and 112 are also connected to network 102. These three clients may be, for example, personal computers or network computers. For purposes of this application, a network computer is any computer coupled to a network, which receives a program or other application from another computer coupled to the network. Distributed data processing system 100 may include additional servers, clients, and other devices not shown.

In the depicted example, servers 104, 114, 116 and 118 provide storage for data from clients 108, 110 and 112. These four servers also provide data, such as boot files, operating system images, and applications to

clients 108, 110 and 112. Clients 108, 110 and 112 are clients to one or all of servers 104, 114, 116 and 118. Support for a particular application being performed on one of clients 108, 110 and 112 may be by one of servers 104, 114, 116 and 118. Additionally servers 104, 114, 116 and 118 may provide backup support for each other. In the event of a server failure, a redundant backup server may be allocated by the network administrator, in which case requests directed to the failed server are routed to the redundant backup server.

In a similar manner, data backup support is provided by storage units 106 and 122 for servers 104, 114, 116 and 118. However, rather than the network administrator allocating a data backup storage unit at each use, data backup allocation is set, and data backup transfer occurs at low usage times, typically after midnight, between any of servers 104, 114, 116 and 118 and storage units 106 and 122.

Prior to the development of Storage Area Networks, each server managed dedicated disk storage, such as Server 104 with disk storage 120. With the advent of a SAN, all disks, such as disks 106, 120, and 122, form a common storage area. When server computers, such as Servers 104, 114, 116, and 118, are added or removed from the SAN, configuration files on each machine that is part of the SAN have to be updated accordingly.

In the depicted example, distributed data processing system 100 may be the Internet, with network 102 representing a worldwide collection of networks and gateways that use the TCP/IP suite of protocols to communicate with one another. At the heart of the Internet is a backbone of high-speed data communication

lines between major nodes or host computers consisting of thousands of commercial, government, education, and other computer systems that route data and messages. Of course, distributed data processing system 100 also may be implemented as a number of different types of networks, such as, for example, an intranet or a local area network.

Figure 1 is intended as an example and not as an architectural limitation for the processes of the present invention. For example, network 102 may use other hardware devices, such as, plotters, optical scanners, and the like in addition or in place of the hardware depicted in Figure 1.

Figure 2 is a block diagram illustrating a data processing system that may be implemented as a server in which the present invention may be implemented. The server in Figure 2 may be server 104 in Figure 1. Data processing system 200 may be a symmetric multiprocessor (SMP) system including a plurality of processors 202 and 204 connected to system bus 206. Alternatively, a single processor system may be employed. Also connected to system bus 206 is memory controller/cache 208, which provides an interface to local memory 209. I/O bus bridge 210 is connected to system bus 206 and provides an interface to I/O bus 212. Memory controller/cache 208 and I/O bus bridge 210 may be integrated as depicted.

Peripheral component interconnect (PCI) bus bridge 214 connected to I/O bus 212 provides an interface to PCI local bus 216. A number of modems 218-220 may be connected to PCI bus 216. Typical PCI bus implementations will support four PCI expansion slots or add-in connectors. Communications links to network

computers 108-112 in Figure 1 may be provided through modem 218 and network adapter 220 connected to PCI local bus 216 through add-in boards.

Additional PCI bus bridges 222 and 224 provide interfaces for additional PCI buses 226 and 228, from which additional modems or network adapters may be supported. In this manner, server 200 allows connections to multiple network computers. A memory mapped graphics adapter 230 and hard disk 232 may also be connected to I/O bus 212 as depicted, either directly or indirectly.

Those of ordinary skill in the art will appreciate that the hardware depicted in Figure 2 may vary. For example, other peripheral devices, such as optical disk drives and the like, also may be used in addition to or in place of the hardware depicted. The depicted example is not meant to imply architectural limitations with respect to the present invention.

The data processing system depicted in Figure 2 may be, for example, an IBM RISC/System 6000, a product of International Business Machines Corporation in Armonk, New York, running the Advanced Interactive Executive (AIX) operating system. AIX is one of the operating systems supported by the current invention. Other supported operating systems are Solaris from Sun Microsystems and Windows NT from Microsoft.

Figure 3 is a block diagram illustrating a data processing system that may be implemented as a client in which the present invention may be implemented. Data processing system 300 is an example of a client computer. Data processing system 300 employs a peripheral component interconnect (PCI) local bus architecture. Although the depicted example employs a PCI bus, other bus

architectures, such as Micro Channel and ISA, may be used. Processor 302 and main memory 304 are connected to PCI local bus 306 through PCI bridge 308. PCI bridge 308 may also include an integrated memory controller and cache memory for processor 302. Additional connections to PCI local bus 306 may be made through direct component interconnection or through add-in boards. In the depicted example, local area network (LAN) adapter 310, SCSI host bus adapter 312, and expansion bus interface 314 are connected to PCI local bus 306 by direct component connection. In contrast, audio adapter 316, graphics adapter 318, and audio/video adapter (A/V) 319 are connected to PCI local bus 306 by add-in boards inserted into expansion slots. Expansion bus interface 314 provides a connection for a keyboard and mouse adapter 320, modem 322, and additional memory 324. In the depicted example, SCSI host bus adapter 312 provides a connection for hard disk drive 326, tape drive 328, CD-ROM drive 330, and digital video disc read only memory drive (DVD-ROM) 332. Typical PCI local bus implementations will support three or four PCI expansion slots or add-in connectors.

An operating system runs on processor 302 and is used to coordinate and provide control of various components within data processing system 300 in Figure 3. The operating system may be a commercially available operating system, such as OS/2, which is available from International Business Machines Corporation. "OS/2" is a trademark of International Business Machines Corporation.

Those of ordinary skill in the art will appreciate that the hardware in Figure 3 may vary depending on the implementation. For example, other peripheral devices,

such as optical disk drives and the like, may be used in addition to or in place of the hardware depicted in Figure 3. The depicted example is not meant to imply architectural limitations with respect to the present invention.

In a preferred embodiment, the present invention controls the task of configuring computing resources connected to a SAN with the use of operating systems, such as, for example, Solaris™, AIX™, and Windows NT™. However, the present invention is not limited to these particular operating system environments and may be implemented with other operating systems as well.

In one embodiment of the present invention, the process involves the development and upkeep of a single configuration file, which may be combined with an "intelligent" script, which places configuration files into the appropriate directories on each SAN attached machine. Within the SAN environment, key configuration files must be stored on each server, such as, for example, server 200 in Figure 2, attached to a network, such as, for example, network 100 in Figure 1. The purpose of these configuration files is to define, for each computational resource, the storage locations which each computational resource may be authorized to use from a storage pool. After these storage locations are identified, all other non-defined locations are invisible to that particular computational resource.

A server may connect to a SAN Fibre Channel by way of a special adapter called the Host Bus Adapter (HBA). Each HBA may have a unique identifier called the World Wide Name (WWN), similar to the way each ethernet card or each token-ring card may have an individualized address. In addition to a switch connection through the HBA, there

is a configuration file, such as, for example, a sd.conf file or a st.conf file, which may specify which disk locations or tape locations the server may access. Each server connecting to the SAN must have such a file. A system administrator may choose to create a unique file for each server which may contain only the entries for that server, or the system administrator may create one master file which contains all entries for all servers. Since each HBA has a unique identifier, there is no possibility of confusing one HBA for another. Furthermore, by having one master configuration file, the system administrator is providing a form of documentation for the entire SAN environment which will help prevent the system administrator from mistakenly assigning one operating system file to a server designed for a second different operating system.

Therefore, the present invention provides a method and system for simplifying and automating the administration of such SANs. Although the current invention normally affects only server machines, if any client machines, such as, for example, client 300 in **Figure 3**, that are also part of the SAN, the client configuration files may also have to be updated when the SAN for the server the clients are connected to is changed in any way. Once a SAN is to be configured using the appropriate configuration file, and once the configuration file name is identified, the present invention takes over the process and automatically configures the server and each client connected to the SAN with the appropriate operating system. The operating system used by the server connected to the SAN is determined. Available operating systems may be, for example, Solaris™, AIX™, Windows NT™, and the like.

Then the server is automatically configured based on the operating system of the server. If it is determined that the server's operating system is not supported by the SAN, an error message may be displayed which indicates that the operating system is not recognized.

Figure 4 is an exemplary illustration of a network containing a storage area network (SAN) in accordance with a preferred embodiment of the present invention. In a preferred embodiment of the present invention, the SAN is made up of one server, although a plurality of servers may be connected to the SAN, in an alternate embodiment, without departing from the spirit and the scope of the present invention.

In this example, network 402 may be made up of several SANs, such as, for example, SAN 404, SAN 406, and SAN 408. Each SAN contained within network 402 is connected to the network via connections, 430, 432, and 434, respectively. Contained within each SAN is a server, such as, for example, server 410 connected to SAN 404 via connection 428. Each server may also contain a number of client computers, such as, for example, clients 412, 414, 416 and 418 connected to server 410. Each client is connected to server via connections, 420, 422, 424 and 426, respectively.

Figure 5 is a flowchart illustrating the automatic configuration of a server within each SAN in accordance with a preferred embodiment of the invention. In this example, the operation begins by opening the file containing information about all members of the SAN. The file containing information about all members of the SAN may be located in the same directory where the SAN configuration files are located. The file containing

information about all members of the SAN may contain one record for each server that is to receive the configuration file(s). Each record in the file may contain two items: the server Internet Protocol (IP) address and the server operating system.

In this example, opening the file may either be done by including the file name on the command line or by interacting with the user. A determination is made as to whether or not the configuration file name for the server is given on the file command line (step 500). If the configuration file name for the server is given on the file command line (step 500:YES), then the specified file is opened for processing (step 502). If no file name is given on the file command line (step 500:NO), then the user is prompted to enter a file name (step 504) and the named configuration file is opened (step 506). A record for a server is then received (step 508).

A determination is then made as to whether or not the server operating system is Solaris (step 510). If the operating system is Solaris (step 510:YES), then the server configuration is performed based on Solaris (step 512). If the operating system is not Solaris (step 510:NO), then there is a determination as to whether or not the server operating system is AIX (step 514). If the operating system is AIX (step 514:YES), then the server configuration is performed based on AIX (step 516). If the operating system is not AIX (step 514:NO), then there is a determination as to whether or not the server operating system is Windows NT (step 518). If the operating system is not Windows NT (step 518:NO), then an error message may be conveyed indicating the operating system is not supported (step 522). This message may be

printed or displayed. If the operating system is Windows NT (step 518:YES), then the server is reconfigured based on Windows NT (step 520).

Once processing the current server is finished, a determination is made as to whether or not there is another server available for processing (step 524). If it is determined that there is not another server available for processing (step 524:NO), then the operation terminates. If it is determined that there is another server available for processing (step 524:YES), the operation returns to step 508 to receive the next record for the next server until all servers are processed. Although the exact commands to carry out the configuration vary slightly based on the operating system, the logical steps in the operation may be the same.

Figure 6 is a flowchart illustrating the transfer of configuration files for an individual server within a SAN in accordance with a preferred embodiment of the invention. There may be two types of files, such as, for example, an "sd" type configuration file for the assignment of disk storage space to a participating server and an "st" type configuration file assigning tape drive storage access to a participating server.

In this example, the operation begins by determining as to whether or not there is a "sd" type configuration file (step 600). If it is determined that there is not a "sd" type configuration file available for copying (step 600:NO), then a determination is made as to whether or not there is a "st" type configuration file available for copying (step 610). If it is determined that there is a "sd" type configuration file available for copying (step

600:YES), then the copy command understood by the operating system is issued to copy the "sd" type configuration file to the appropriate directory (step 602). Then a determination is made as to whether or not the copying of the "sd" type configuration file was successful (step 604). If the copying of the "sd" type configuration file was successful (step 604:YES), then a message is output giving the details of the file copy results (step 606). The successful information message may be appended to the file /SAN.copy.config.file.log which is located on the server where this configuration program script resides. If the copying of the "sd" type configuration file fails for any reason (step 604:NO), an appropriate error message is displayed indicating the failure (step 608). The error message may be printed or visually displayed on a main console connected to the server.

Returning to step 610, whether a message is appended giving details of the "sd" type configuration file copy results or an error message is displayed indicating that the "sd" configuration file copy failed, then it is determined as to whether or not there is an "st" type configuration file available for copying (step 610). If it is determined that there is not a "st" type configuration file available for copying (step 610:NO), then the operation terminates. If it is determined that there is a "st" type configuration file to be copied (step 610:YES), then a copy command appropriate for the operating system is issued and the "st" type configuration file is copied (step 612). Then a determination is made as to whether or not the copying of the "st" type configuration file was successful (step

614). If the copying of the "st" type configuration file failed for any reason (step 614:NO), then an error message is displayed indicating the failure (step 618) and thereafter the operation terminates. The error message may be printed or visually displayed on a main console connected to the server. If the copying of the "st" type configuration file was successful (step 614:YES), then a message is output giving the details of the file copy results (step 616) and thereafter the operation terminates. The successful information message is appended to the file /SAN.copy.config.file.log which is located on the server where this configuration program script resides.

Figures 7A-7D is an exemplary shell script for updating configuration files in accordance with a preferred embodiment of the invention. In this example, section 700 contains comments providing program documentation. Section 702 opens a file containing the list of servers either through a command line argument or through user interaction. This code corresponds to steps 500-508 in Figure 5.

The program reads the next server to be processed in Section 704. This corresponds to step 510 in Figure 5. The type of operating system used by the server is determined and the "sd" configuration file and "st" type configuration file are copied, as necessary. If the operating system is Solaris, then Section 706 copies the "sd" type configuration file. This corresponds to steps 600-608 in Figure 6. If a "st" type configuration file has to be copied, this is done by the code in Section 708 corresponding to steps 610-618 in Figure 6.

Other operating systems are handled in a similar manner. In this example, section 710 is for the AIX operating system and Section 712 is for the Windows NT operating system. As shown in Section 714, any unsupported operating system results in the display of an error message. After this is complete, control returns to Section 704 where the next server is processed. The entire operation halts when there are no more servers to be processed.

The operation describes a process for automatically updating all configuration files for a server in a Storage Area Network based on a single master file which makes it much easier to handle changes to a SAN in an efficient and error free manner. When a server is added or removed from the SAN, this change only needs to be entered once in the master configuration file. The script described in Figures 5-6 and the code given in Figure 7 automatically propagates these changes throughout the SAN. Any discrepancies in carrying out these changes may be logged for review by a system administrator.

It is important to note that while the present invention has been described in the context of a fully functioning data processing system, those of ordinary skill in the art will appreciate that the processes of the present invention are capable of being distributed in the form of a computer readable medium of instructions and a variety of forms and that the present invention applies equally regardless of the particular type of signal bearing media actually used to carry out the distribution. Examples of computer readable media include recordable-type media such as a floppy disc, a

hard disk drive, a RAM, and CD-ROMs and transmission-type media such as digital and analog communications links.

The description of the present invention has been presented for purposes of illustration and description, but is not intended to be exhaustive or limited to the invention in the form disclosed. Many modifications and variations will be apparent to those of ordinary skill in the art. The embodiment was chosen and described in order to best explain the principles of the invention, the practical application, and to enable others of ordinary skill in the art to understand the invention for various embodiments with various modifications as are suited to the particular use contemplated.